DAVID R. EBERHART (S.B. #195474)
deberhart@omm.com
JAMES K. ROTHSTEIN (S.B. #267962)
jrothstein@omm.com
O'MELVENY & MYERS LLP
Two Embarcadero Center
28th Floor
San Francisco, California 94111-3823
Telephone: +1 415 984 8700
Facsimile: +1 415 984 8701

Attorneys for Plaintiffs
ELASTICSEARCH, INC. and
ELASTICSEARCH B.V.

UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

ELASTICSEARCH, INC., a Delaware corporation, ELASTICSEARCH B.V., a Dutch corporation,

Plaintiffs,

v.

FLORAGUNN GmbH, a German corporation,

Defendant.

Case No. 4:19-cv-05553-YGR

FIRST AMENDED COMPLAINT

1. COPYRIGHT INFRINGEMENT, 17 U.S.C. § 101 *ET SEQ*.
2. CONTRIBUTORY COPYRIGHT INFRINGEMENT

JURY TRIAL DEMAND

INTRODUCTION

1. Elasticsearch, Inc. and elasticsearch B.V. (collectively "Elastic") bring this action to remedy floragunn GmbH's ("floragunn") knowing and willful infringement of Elastic's copyright in the source code for Elastic's X-Pack software.

2. Elastic is the creator of the Elastic Stack suite of products that is centered on the popular and powerful Elasticsearch search and analytics engine. Leading companies and organizations like Cisco Systems, Facebook, and NASA's Jet Propulsion Laboratory at the California Institute of Technology use and depend upon Elasticsearch.

3. Elastic offers a set of features, known as X-Pack, that enhance and extend the Elastic Stack suite of products. In keeping with its longstanding commitment to openness, Elastic made the source code for X-Pack publicly available in 2018 subject to certain restrictions. Among other rights, Elastic clearly reserved commercial rights in X-Pack and its derivative works.

4. floragunn markets and distributes Search Guard, a plug-in for Elasticsearch that is intended to compete with the security features of X-Pack. Yet instead of fairly competing with Elastic and developing Search Guard with its own resources, floragunn copied multiple and critical portions of Elastic's X-Pack proprietary security source code into its Search Guard product.

5. One particularly large incident of copying occurred just one month after Elastic publicly opened X-Pack's source code. Elastic's examination of floragunn's then-publicly available code on GitHub demonstrates that, at that time, floragunn made dramatic alterations to Search Guard in a single, massive effort that it released—contrary to common programming practice and floragunn's own past practices—without any substantive explanation.

6. But this was not the beginning or the end of floragunn's infringement. Elastic has now discovered evidence that floragunn's copying and creation of derivative works from Elastic's code extends back to at least 2015. Because Elastic had released that code only in binary form, moreover, it was necessary for floragunn to intentionally decompile that code to enable the copying and creation of derivative works. Furthermore, Elastic has now determined that floragunn copied and created derivative works not only from Elastic's X-Pack code containing

security features for its Elasticsearch software—floragunn also copied and created derivative works from Elastic's X-Pack code containing security features for its Kibana software.

7. Once floragunn copied Elastic's code, it then licensed its infringing Search Guard software to corporations and institutions, including a significant number that are located in the Northern District of California. These acts by floragunn induced further infringements of Elastic's copyrights by those third parties, including through products and services offered by those third parties. For example, Elastic has now determined that the Amazon Elasticsearch Service and Open Distro for Elasticsearch from Amazon.com, Inc. and Amazon Web Services, Inc., as well as Rackspace US, Inc.'s ObjectRocket for Elasticsearch and IBM Corporation's Cloud Databases for Elasticsearch include and/or recently included infringing code.

8. floragunn's response to Elastic's infringement claims is also consistent with copyright infringement by floragunn. After the commencement of this lawsuit, Elastic issued takedown notices under the Digital Millennium Copyright Act ("DMCA") to websites that were hosting floragunn's infringing code. Those websites removed floragunn's code in response to Elastic's notices. floragunn had the right to issue counter notifications to those websites to assert that floragunn was, contrary to Elastic's notices, the owner of the copyright to the code in question. But Elastic has seen no such notices from floragunn because, on information and belief, floragunn issued no such notices. But what floragunn did do is telling: it moved hosting for downloads of its infringing code to a hosting provider that expressly advertises that it will not comply with the DMCA. The hosting provider's website states: "Purchasing USA-based hosting for a site that is not legal to be run in America is not a sensible thing to do. Offshore hosting can be helpful for less scrupulous businesses who wish to bypass local laws or regulations, particularly for issues like copyright law, which is also known as no DMCA hosting."

9. floragunn's unauthorized reproduction, creation of derivative works, and distribution of Elastic's copyrighted software code constitutes copyright infringement under 17 U.S.C. § 101 *et seq*. floragunn is further liable for contributory copyright infringement because it intentionally induced Search Guard users and third parties that integrate Search Guard code into

their own products and services to infringe Elastic's copyrights. Elastic seeks injunctive and monetary relief to the maximum extent permitted by law.

PARTIES

10. Plaintiff Elasticsearch, Inc. is incorporated in Delaware; it has its principal place of business in Mountain View, California. Plaintiff elasticsearch B.V. is incorporated in the Netherlands.

11. Defendant floragunn is a German company with a principal place of business in Berlin, Germany.

12. Elastic is aware that there are third party users and adopters of floragunn's infringing Search Guard product and code. Elastic may seek leave to amend to add those third parties as defendants following discovery from floragunn.

JURISDICTION AND VENUE

13. Elastic's claims for copyright infringement arise under the Copyright Act of 1976, 17 U.S.C. § 101 *et seq*.

14. This Court has original subject matter jurisdiction of this action under 28 U.S.C. §§ 1331 and 1338.

15. This Court has specific personal jurisdiction over floragunn because, among other reasons, floragunn has extensively offered and distributed its infringing product containing Elastic's copyrighted material to companies in California and purposefully committed within California the acts upon which Elastic's claims arise. Additionally, to the extent floragunn has committed the illegal acts described herein outside of California, it did so knowing and intending that such acts would cause injury to Elastic at its principal place of business within California.

16. Venue is proper in the Northern District of California under 28 U.S.C. § 1391(b)(2) and 1391(c)(3) because a substantial part of the events or omissions giving rise to the claims alleged in this complaint occurred in this judicial district.

INTRADISTRICT ASSIGNMENT

17. Because this action arises from Elastic's assertion of its intellectual property rights, Northern District of California Local Rule 3.2(c) excludes this action from the division-specific venue rule and subjects this action to assignment on a district-wide basis.

THE ELASTIC STACK AND X-PACK SOFTWARE

18. Elastic produces a core suite of search and analytics products known as Elastic Stack (formerly known as ELK Stack). The Elastic Stack consists of Elasticsearch, Logstash, Kibana, and Beats. Elasticsearch is a search and analytics engine. Logstash is a server-side data processing pipeline that ingests data from multiple sources simultaneously, transforms it, and then sends it to a "stash" like Elasticsearch. Kibana lets users visualize data with charts and graphs in Elasticsearch. Beats is a family of "data shipper" software that collects and centralizes data that feeds into the other products in Elastic Stack.

19. X-Pack is a set of add-on features to Elastic's core Elastic Stack suite of products. X-Pack includes security, altering, monitoring, reporting, and other add-ons to Elasticsearch, Kibana, and other products in the Elastic Stack. The predecessor to much of X-Pack was known as Shield. Elastic refers to Shield and X-Pack collectively herein as "X-Pack."

20. Elastic has a longstanding commitment to opening the source code underlying many of its products in order to facilitate building a community that helps improve and advance Elastic's products to produce the best software possible. When Elastic releases the source code for its software, it does so under clearly delineated conditions.

21. In late April 2018, Elastic opened the source code for version 6.2.x of X-Pack. Elastic made the code available on Elastic's public GitHub code repository for users to inspect, contribute, create issues, and open pull requests, all pursuant to the "Elastic License." Elastic has released the source code for subsequent versions of X-Pack on GitHub, also under the "Elastic License."

22. The Elastic License did not grant to floragunn or any other party the right to create copies or prepare derivative works for use in any production capacity. And to the extent floragunn acquired any rights pursuant to the Elastic License, those rights terminated immediately and

automatically by virtue of floragunn's breaches as described herein. Nor did any license applicable to earlier versions of X-Pack and/or Shield provide floragunn the right to create copies or prepare derivative works for use in any production capacity.

FLORAGUNN'S INFRINGEMENT OF ELASTIC'S COPYRIGHTS IN X-PACK

23. floragunn markets and distributes Search Guard, a plug-in for Elasticsearch that offers features similar to the security features that Elastic offers in X-Pack. floragunn makes the source code for Search Guard available for review and inspection on its GitLab repositories under several different license agreements. (Before the commencement of this lawsuit, floragunn made the source code for Search Guard available through GitHub repositories.)

24. Search Guard is available as a "Community Edition" for free for certain uses, but floragunn charges customers for Enterprise and Compliance editions of Search Guard. floragunn prohibits users from, among other things, taking features from the Enterprise or Compliance editions of Search Guard into production without purchasing a license. In fact, floragunn explicitly warns its users that doing so "is illegal" and "can lead to serious legal consequences, which can bring more harm and costs to a company . . . ."

25. Elastic is informed and believes, and, on that basis, alleges that after Elastic made the source code for X-Pack version 6.2.x publicly available, floragunn accessed significant portions of at least the version 6.2.x code, copied and/or created derivative works from that code, and reproduced and distributed it in the code for Search Guard.

26. On June 7, 2018, just over one month after Elastic made the source code for X-Pack version 6.2.x publicly available under the Elastic License, floragunn made a sudden and very large change to the Search Guard code. This change comprised 244 additions and 145 deletions of code. Many of these changes involved the wholesale copying of the X-Pack code that Elastic opened little over a month before.

27. A significant portion of floragunn's copying centered on the Document Level Security ("DLS") features in Elastic's X-Pack code. As the name would suggest, DLS allows an X-Pack customer to apply security settings to particular documents in the database.

28. As part of its June 7, 2018, changes, floragunn copied the implementations of at least two methods from the X-Pack code, getLiveDocs and numDocs, from the file DocumentSubsetReader.java.

29. A comparison of Elastic's implementation of getLiveDocs in X-Pack and floragunn's implementation of method getLiveDocs in Search Guard shows that floragunn's implementation is substantively identical to Elastic's implementation:

Elastic's Implementation of getLiveDocs:

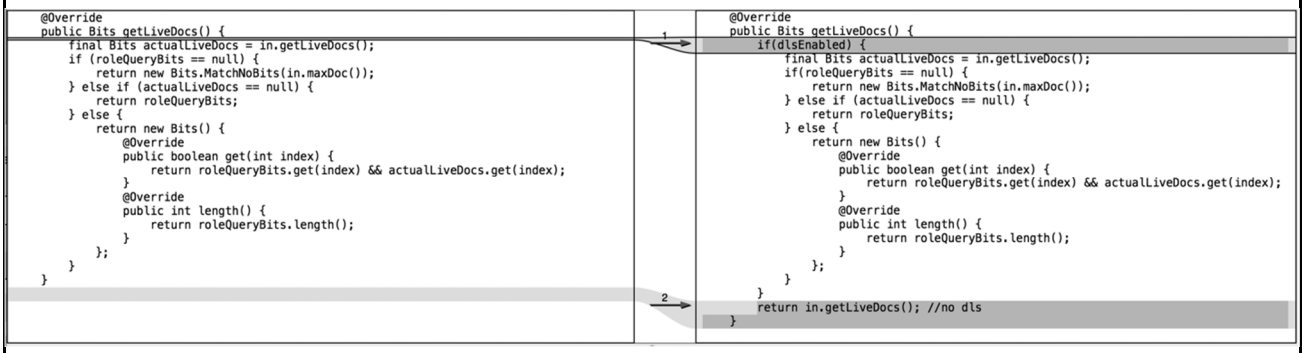
```
@Override
   public Bits getLiveDocs() {
      final Bits actualLiveDocs = in.getLiveDocs();
      if (roleQueryBits == null) {
         // If we would a <code>null</code> liveDocs then that would mean that no
docs are marked as deleted,
         // but that isn't the case. No docs match with the role query and therefor all
docs are marked as deleted
         return new Bits.MatchNoBits(in.maxDoc());
      } else if (actualLiveDocs == null) {
         return roleQueryBits;
      } else {
         // apply deletes when needed:
         return new Bits() {

            @Override
            public boolean get(int index) {
               return roleQueryBits.get(index) && actualLiveDocs.get(index);
            }

            @Override
            public int length() {
               return roleQueryBits.length();
            }
         };
      }
   }
```

floragunn's Implementation of getLiveDocs:

```
@Override
   public Bits getLiveDocs() {

      if(dlsEnabled) {
         final Bits currentLiveDocs = in.getLiveDocs();

         if(bs == null) {
            return new Bits.MatchNoBits(in.maxDoc());
         } else if (currentLiveDocs == null) {
            return bs;
```

```
            } else {
               return new Bits() {
                  @Override
                  public boolean get(int index) {
                     return bs.get(index) && currentLiveDocs.get(index);
                  }
                  @Override
                  public int length() {
                     return bs.length();
                  }
               };
            }
         }
         return in.getLiveDocs(); //no dls
      }
```

30. By removing comments and superfluous blank lines, and by making variable names consistent, it becomes apparent that the Search Guard code is copied from or is, at least, a derivative work of Elastic's code. (Elastic's code is on the left; floragunn's is on the right.) A larger version of this graphic is attached to this Complaint as Exhibit A.



```
@Override
public Bits getLiveDocs() {
    final Bits actualLiveDocs = in.getLiveDocs();
    if (roleQueryBits == null) {
        return new Bits.MatchNoBits(in.maxDoc());
    } else if (actualLiveDocs == null) {
        return roleQueryBits;
    } else {
        return new Bits() {
            @Override
            public boolean get(int index) {
                return roleQueryBits.get(index) && actualLiveDocs.get(index);
            }
            @Override
            public int length() {
                return roleQueryBits.length();
            }
        };
    }
}
```

```
@Override
public Bits getLiveDocs() {
    if(dlsEnabled) {
        final Bits actualLiveDocs = in.getLiveDocs();
        if(roleQueryBits == null) {
            return new Bits.MatchNoBits(in.maxDoc());
        } else if (actualLiveDocs == null) {
            return roleQueryBits;
        } else {
            return new Bits() {
                @Override
                public boolean get(int index) {
                    return roleQueryBits.get(index) && actualLiveDocs.get(index);
                }
                @Override
                public int length() {
                    return roleQueryBits.length();
                }
            };
        }
    }
    return in.getLiveDocs(); //no dls
}
```

31. Similarly, floragunn's June 7, 2018 commit changed Search Guard's implementation of the method numDocs to be essentially identical to Elastic's implementation in X-Pack. Here is Elastic's implementation, again from the file DocumentSubsetReader.java:

```
   @Override
   public int numDocs() {
      // The reason the implement this method is that numDocs should be
equal to the number of set bits in liveDocs. (would be weird otherwise)
```

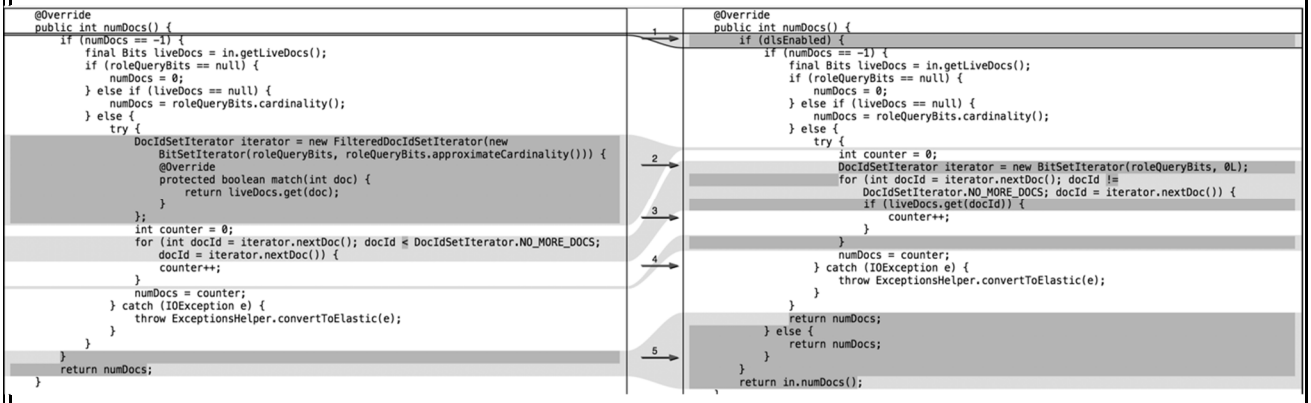
```
        // for the Shield DSL use case this get invoked in the QueryPhase
    class (in core ES) if match_all query is used as main query
        // and this is also invoked in tests.
        if (numDocs == -1) {
            final Bits liveDocs = in.getLiveDocs();
            if (roleQueryBits == null) {
                numDocs = 0;
            } else if (liveDocs == null) {
                numDocs = roleQueryBits.cardinality();
            } else {
                // this is slow, but necessary in order to be correct:
                try {
                    DocIdSetIterator iterator = new FilteredDocIdSetIterator(new
    BitSetIterator(roleQueryBits, roleQueryBits.approximateCardinality())) {
                        @Override
                        protected boolean match(int doc) {
                            return liveDocs.get(doc);
                        }
                    };
                    int counter = 0;
                    for (int docId = iterator.nextDoc(); docId <
    DocIdSetIterator.NO_MORE_DOCS; docId = iterator.nextDoc()) {
                        counter++;
                    }
                    numDocs = counter;
                } catch (IOException e) {
                    throw ExceptionsHelper.convertToElastic(e);
                }
            }
        }
        return numDocs;
    }
```

32. Again, floragunn's June 7, 2018, changes altered Search Guard's implementation of the method numDocs to be substantively identical to Elastic's implementation in X-Pack:

```
    @Override
    public int numDocs() {
        if (dlsEnabled) {
            if (this.numDocs == -1) {
                final Bits currentLiveDocs = in.getLiveDocs();
                if (bs == null) {
                    this.numDocs = 0;
                } else if (currentLiveDocs == null) {
                    this.numDocs = bs.cardinality();
                } else {
                    try {
                        int localNumDocs = 0;
                        DocIdSetIterator it = new BitSetIterator(bs, 0L);
                        for (int doc = it.nextDoc(); doc !=
    DocIdSetIterator.NO_MORE_DOCS; doc = it.nextDoc()) {
                            if (currentLiveDocs.get(doc)) {
                                localNumDocs++;
                            }
```

```
                        }
                        this.numDocs = localNumDocs;
                    } catch (IOException e) {
                        throw ExceptionsHelper.convertToElastic(e);
                    }
                }
                return this.numDocs;
            } else {
                return this.numDocs; // cached
            }
        }
        return in.numDocs();
    }
```

33. Ignoring non-substantive differences in the code (*i.e.*, removing blank lines, conforming variable names, and removing the superfluous "this." in front of certain variables), it is clear that the floragunn code (on the right) is copied from or, at least, a derivative work of the Elastic code (on the left). A larger version of this graphic is attached to this Complaint as Exhibit B.



34. floragunn's June 7, 2018, changes also included several other alterations to Search Guard that mimic X-Pack, including, at least: (1) changing the computation of Search Guard's BitSet from an inferior IndexSearcher to align itself with how X-Pack computes the BitSet; and (2) changing computation of live documents to match the X-Pack implementation.

35. floragunn took efforts to keep its misconduct concealed. For example, the only explanation floragunn provided for the changes it made on June 7 was "Improve dls/fls." This is a strikingly brief explanation in light of the significant changes floragunn had committed to its code base. And such minimal explanation is inconsistent not only with standard computer

programming practices but is also inconsistent with floragunn's explanations accompanying its commits of other code.

36. floragunn's June 7, 2018, changes also lack evidence that floragunn undertook unit testing of the code—yet another absence that is inconsistent with common programming practice and different from floragunn's other public code. This too strongly suggests that floragunn simply copied Elastic's code.

37. Examination of floragunn's Search Guard code reveals that its recent acts of infringement are consistent with a larger and longstanding pattern of misconduct.

38. Code released by floragunn as part of Search Guard in 2016 contains the following commented out—that is, non-functional—code:

```
// "internal:*",
// "indices:monitor/*",
// "cluster:monitor/*",
// "cluster:admin/reroute",
// "indices:admin/mapping/put"
```

39. That code was copied verbatim from the following functional Elastic code in Shield (Elastic's security product that preceded X-Pack) that was released in or before 2015:

```
protected static final Predicate<String> PREDICATE =
new AutomatonPredicate(patterns(
"internal:*",
"indices:monitor/*", // added for marvel
"cluster:monitor/*", // added for marvel
"cluster:admin/reroute", // added for DiskThresholdDecider.DiskListener
"indices:admin/mapping/put" // ES 2.0
MappingUpdatedAction -
updateMappingOnMasterSynchronously
));
```

40. Elastic had not publicly released this source code for Shield at the time of floragunn's copying and/or creation of derivative works from that code. Elastic is informed and believes and, on that basis, alleges that floragunn decompiled Elastic's binaries or otherwise gained access to Elastic's source code to create the copies and/or derivative works referenced in Paragraph 38.

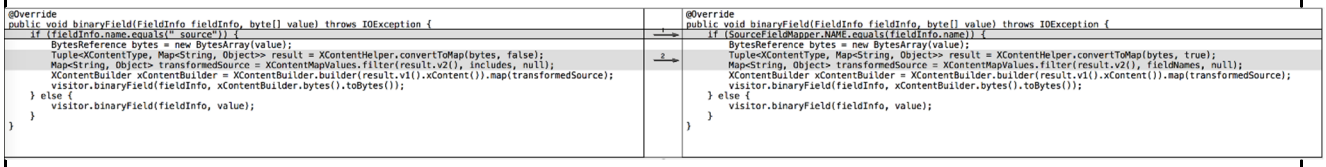
41. Code released by floragunn on June 6, 2016, into the search-guard-module-dlsfls repository for Search Guard contains the following:

```
@Override
public void binaryField(final FieldInfo, final byte[] value) throws IOException {

   if (fieldInfo.name.equals("_source")) {
      final BytesReference bytesRef = new BytesArray(value);
      final Tuple<XContentType, Map<String, Object>> bytesRefTuple =
XContentHelper.convertToMap(bytesRef, false);
      final Map<String, Object> filteredSource =
XContentMapValues.filter(bytesRefTuple.v2(), includes, null);
      final XContentBuilder xBuilder =
XContentBuilder.builder(bytesRefTuple.v1().xContent()).map(filteredSource);
      delegate.binaryField(fieldInfo, xBuilder.bytes().toBytes());
   } else {
      delegate.binaryField(fieldInfo, value);
   }
}
```

42. That code is substantively identical to the following Elastic code that had previously been included in Shield:

```
@Override
public void binaryField(FieldInfo, byte[] value) throws IOException {
   if (SourceFieldMapper.NAME.equals(fieldInfo.name)) {
      // for _source, parse, filter out the fields we care about, and serialize back
downstream
      BytesReference bytes = new BytesArray(value);
      Tuple<XContentType, Map<String, Object>> result =
XContentHelper.convertToMap(bytes, true);
      Map<String, Object> transformedSource = XContentMapValues.filter(result.v2(),
fieldNames, null);
      XContentBuilder =
XContentBuilder.builder(result.v1().xContent()).map(transformedSource);
      visitor.binaryField(fieldInfo, xContentBuilder.bytes().toBytes());
   } else {
      visitor.binaryField(fieldInfo, value);
   }
}
```

43. Ignoring non-substantive differences in the code, it is clear that the floragunn code (on the left) is copied from or, at least, a derivative work of the Elastic code (on the right). A larger version of this graphic is attached to this Complaint as Exhibit C.



```
@Override
public void binaryField(FieldInfo fieldInfo, byte[] value) throws IOException {
    if (fieldInfo.name.equals("_source")) {
        BytesReference bytes = new BytesArray(value);
        Tuple<XContentType, Map<String, Object>> result = XContentHelper.convertToMap(bytes, false);
        Map<String, Object> transformedSource = XContentMapValues.filter(result.v2(), includes, null);
        XContentBuilder xContentBuilder = XContentBuilder.builder(result.v1().xContent()).map(transformedSource);
        visitor.binaryField(fieldInfo, xContentBuilder.bytes().toBytes());
    } else {
        visitor.binaryField(fieldInfo, value);
    }
}
```

```
@Override
public void binaryField(FieldInfo fieldInfo, byte[] value) throws IOException {
    if (SourceFieldMapper.NAME.equals(fieldInfo.name)) {
        BytesReference bytes = new BytesArray(value);
        Tuple<XContentType, Map<String, Object>> result = XContentHelper.convertToMap(bytes, true);
        Map<String, Object> transformedSource = XContentMapValues.filter(result.v2(), fieldNames, null);
        XContentBuilder xContentBuilder = XContentBuilder.builder(result.v1().xContent()).map(transformedSource);
        visitor.binaryField(fieldInfo, xContentBuilder.bytes().toBytes());
    } else {
        visitor.binaryField(fieldInfo, value);
    }
}
```

44. Elastic had not publicly released this source code for Shield at the time of floragunn's copying and/or creation of derivative works from that code. Elastic is informed and believes and, on that basis, alleges that floragunn decompiled Elastic's binaries or otherwise gained access to Elastic's source code to create the copies and/or derivative works referenced in Paragraph 41.

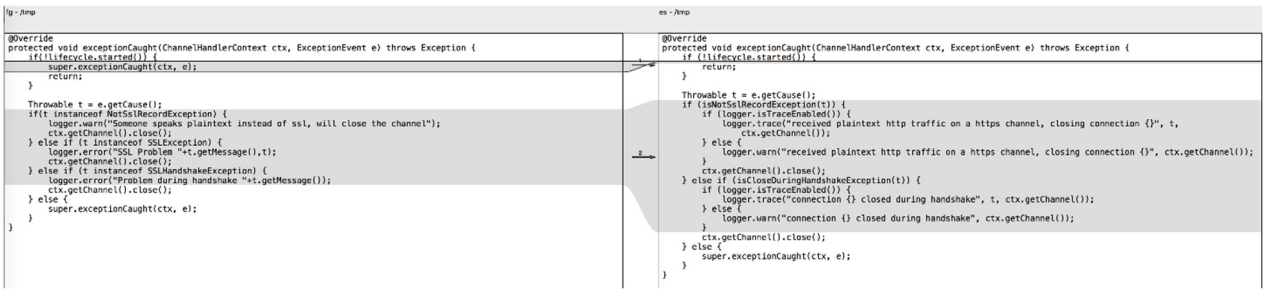
45. Infringement by floragunn is evident in additional code in the ShieldNettyHttpServerTransport file. Code released by floragunn on December 10, 2016 as part of the Search Guard SearchGuardSSLNettyHttpServerTransport file contains the following content:

```
@Override
protected void exceptionCaught(ChannelHandlerContext ctx, ExceptionEvent e) throws
Exception {
  if(this.lifecycle.started()) {

    final Throwable cause = e.getCause();
    if(cause instanceof NotSslRecordException) {
      logger.warn("Someone speaks plaintext instead of ssl, will close the channel");
      ctx.getChannel().close();
      return;
    } else if (cause instanceof SSLException) {
      logger.error("SSL Problem "+cause.getMessage(),cause);
      ctx.getChannel().close();
      return;
    } else if (cause instanceof SSLHandshakeException) {
      logger.error("Problem during handshake "+cause.getMessage());
      ctx.getChannel().close();
      return;
    }
  }

  super.exceptionCaught(ctx, e);
}
```

46. That code is substantively identical to the following Elastic code included in the binary of Elastic Shield released June 24, 2015:

```
@Override
protected void exceptionCaught(ChannelHandlerContext ctx, ExceptionEvent e) throws
Exception {
   if (!lifecycle.started()) {
      return;
   }

   Throwable t = e.getCause();
   if (isNotSslRecordException(t)) {
      if (logger.isTraceEnabled()) {
         logger.trace("received plaintext http traffic on a https channel, closing connection
{}", t, ctx.getChannel());
      } else {
         logger.warn("received plaintext http traffic on a https channel, closing connection
{}", ctx.getChannel());
      }
      ctx.getChannel().close();
   } else if (isCloseDuringHandshakeException(t)) {
      if (logger.isTraceEnabled()) {
         logger.trace("connection {} closed during handshake", t, ctx.getChannel());
      } else {
         logger.warn("connection {} closed during handshake", ctx.getChannel());
      }
      ctx.getChannel().close();
   } else {
      super.exceptionCaught(ctx, e);
   }
}
```

47. Ignoring non-substantive differences in the code, it is clear that the floragunn code (on the left) is copied from or, at least, a derivative work of the Elastic code (on the right). A larger version of this graphic is attached to this Complaint as Exhibit D.



48. Elastic had not publicly released this source code for Shield at the time of floragunn's copying and/or creation of derivative works from that code. Elastic is informed and

believes and, on that basis, alleges that floragunn decompiled Elastic's binaries or otherwise gained access to Elastic's source code to create the copies and/or derivative works referenced in Paragraph 45.

FLORAGUNN'S INFRINGEMENT OF ELASTIC'S COPYRIGHTS IN THE X-PACK KIBANA PLUGIN

49. Subsequent investigation has also revealed floragunn's copying and/or creation of derivative works from code from the X-Pack plugin for Elastic's Kibana product. The infringed code that Elastic has identified comes from the X-Pack Kibana elements Get Next URL, Saved Objects Client, AngularJS Management Screens, callWithRequestFactory, and fetchAllFromScroll. In addition to the examples below, Elastic has identified copying and/or creation of derivative works in April 5, 2017, August 6, 2017, and June 4, 2019 commits to Search Guard.

50. As one example, Search Guard code released by floragunn on March 31, 2018 in get_next_url.js contains the following code:

```
const {query, hash} = parse(currentUrl, true);
if (!query.nextUrl) {
  return `${basePath}/`;
}
const { protocol, hostname, port, pathname } = parse(query.nextUrl);
if (protocol || hostname || port) {
  return `${basePath}/`;
}
if (!String(pathname).startsWith(basePath)) {
  return `${basePath}/`;
}
return query.nextUrl + (hash || '');
```

51. That code closely mirrors the following code that Elastic included in a bug fix to the X-Pack Kibana plugin in parse_next.js on April 4, 2017:

```
const { query, hash } = parse(href, true);
if (!query.next) {
  return `${basePath}/`;
}
const { protocol, hostname, port, pathname } = parse(query.next);
if (protocol || hostname || port) {
  return `${basePath}/`;
}
```

```
if (!String(pathname).startsWith(basePath)) {
  return `${basePath}/`;
}
return query.next + (hash || '');
```

52. Ignoring non-substantive differences in the code, it is clear that the floragunn code in Paragraph 50 is copied from, or at least a derivative work of, the Elastic code in Paragraph 51.

53. As another example, Search Guard code released by floragunn on October 28, 2018 in get_next_url.js contains the following code:

```
const {query, hash} = parse(currentUrl, true, true);
if (!query.nextUrl) {
   return `${basePath}/`;
}
const { protocol, hostname, port, pathname } = parse(query.nextUrl, false, true);
if (protocol !== null || hostname !== null || port !== null) {
   return `${basePath}/`;
}
if (!String(pathname).startsWith(basePath)) {
   return `${basePath}/`;
}
return query.nextUrl + (hash || '');
```

54. That code is nearly identical to the following code that Elastic included in a bug fix to the X-Pack Kibana plugin in parse_next.js on January 28, 2018:

```
const { query, hash } = parse(href, true);
if (!query.next) {
  return `${basePath}/`;
}
const { protocol, hostname, port, pathname } = parse(
  query.next,
  false,
  true
);
if (protocol !== null || hostname !== null || port !== null) {
  return `${basePath}/`;
}
if (!String(pathname).startsWith(basePath)) {
  return `${basePath}/`;
}
return query.next + (hash || '');
```

55. Ignoring non-substantive differences in the code, it is clear that the floragunn code in Paragraph 53 is copied from, or at least a derivative work of, the Elastic code in Paragraph 54.

56. As another example, Search Guard code released by floragunn on August 30, 2019 in call_with_request_factory.js contains the following code:

```
import { once } from 'lodash';
import { elasticsearchSignalsPlugin } from '../elasticsearch_signals_plugin';
import { CLUSTER } from '../../../../utils/signals/constants';
const callWithRequest = once((server) => {
  const { callWithRequest } =
server.plugins.elasticsearch.createCluster(CLUSTER.ALERTING, {
    plugins: [elasticsearchSignalsPlugin]
  });
  return callWithRequest;
});
export const callWithRequestFactory = (server, request) =>
  (...rest) => callWithRequest(server)(request, ...rest);
```

57. That code is nearly identical to the following Elastic code that occurs multiple places within the X-Pack Kibana plugin, including in a February 28, 2019 commit to call_with_request_factory.js reproduced here:

```
import { once } from 'lodash';
import { elasticsearchJsPlugin } from '../elasticsearch_js_plugin';
const callWithRequest = once((server) => {
  const config = { plugins: [ elasticsearchJsPlugin ] };
  const cluster = server.plugins.elasticsearch.createCluster('watcher', config);

  return cluster.callWithRequest;
});
export const callWithRequestFactory = (server, request) => {
  return (...args) => {
    return callWithRequest(server)(request, ...args);
  };
};
```

58. Ignoring non-substantive differences in the code, it is clear that the floragunn code in Paragraph 56 is copied from, or at least a derivative work of, the Elastic code in Paragraph 57.

59. As one more example, Search Guard code released by floragunn also on August 30, 2019 in fetch_all_from_scroll.js contains the following code:

```
import { ES_SCROLL_SETTINGS } from '../../../../utils/signals/constants';
export function fetchAllFromScroll(response, callWithRequest, allHits = []) {
  const { _scroll_id: scrollId, hits: { hits = [] } } = response;
  if (hits.length) {
    allHits.push(...hits);
    return callWithRequest('scroll', {
      body: {
        scroll: ES_SCROLL_SETTINGS.KEEPALIVE,
        scroll_id: scrollId
      }
    }).then(_response => fetchAllFromScroll(_response, callWithRequest, allHits));
  }
  return Promise.resolve(allHits);
```

```
}
```

60. That code very closely mirrors the following Elastic code included in an April 6, 2017 commit to fetch_all_from_scroll.js reproduced here:

```
import { get } from 'lodash';
import { ES_SCROLL_SETTINGS } from '../../../common/constants';
export function fetchAllFromScroll(response, callWithRequest, hits = []) {
  const newHits = get(response, 'hits.hits', []);
  const scrollId = get(response, '_scroll_id');
  if (newHits.length > 0) {
    hits.push(...newHits);
    return callWithRequest('scroll', {
      body: {
        scroll: ES_SCROLL_SETTINGS.KEEPALIVE,
        scroll_id: scrollId
      }
    })
      .then(innerResponse => {
        return fetchAllFromScroll(innerResponse, callWithRequest, hits);
      });
  }
  return Promise.resolve(hits);
}
```

61. Ignoring non-substantive differences in the code, it is clear that the floragunn code in Paragraph 59 is copied from, or at least a derivative work of, the Elastic code in Paragraph 60.

62. A comment made by a floragunn programmer in a June 4, 2019 Search Guard commit provides further proof of copying because it indicates that the programmer did not understand the reason that a variable in the source code was formatted in a particular way. Programmers must choose the format or formats for the names of variables in their source code. This choice is often more than stylistic, because it can affect compatibility with other programs and operating systems. One such format is "snake case" where a programmer replaces all spaces with "_." Accordingly, a variable named "first variable" would be written in snake case as "first_variable."

63. Elastic's X-Pack Kibana plugin formats variables in "snake case" to remain compatible with the X-Pack plugin for Elasticsearch—but, on information and belief, that reason for formatting variables in "snake case" is not present for the infringing Search Guard code. Although the reason for use of "snake case" is absent, floragunn's infringing code also uses "snake case" for the "bulk_create" variable. But a floragunn programmer left a comment in the

infringing Search Guard code noting that s/he could not determine why the code used "snake case" for the "bulk_create" variable, writing: "@todo Why the snake case here? What do our permissions look like."

FLORAGUNN MARKETED THE INFRINGING WORK IN THE NORTHERN DISTRICT OF CALIFORNIA

64. floragunn's Search Guard product directly competes with the security features in Elastic's X-Pack and X-Pack Kibana plugin.

65. Elastic is informed and believes, and, on that basis alleges that floragunn knew that Elastic had its principal place of business in the Northern District of California.

66. floragunn maintains significant and ongoing commercial ties to the Northern District of California. The industry that provides security features for Elastic Stack is very small, and, Elastic is informed and believes, is composed of at most six companies. Despite the small number of companies providing security features for Elastic Stack, the customer base for Elastic Stack security features is broad. floragunn boasts of a "global customer base," including "many of the tech giants." Due to the prominence of the technology industry in the Northern District of California, many of these companies are headquartered in, maintain offices in, or do significant business in the Northern District of California.

67. Before Elastic commenced this lawsuit, floragunn hosted its infringing source code on a website run by GitHub, Inc. GitHub, Inc. is headquartered and maintains its principal place of business in San Francisco, California, within the Northern District of California. floragunn currently hosts infringing source code through GitLab Inc., a company also headquartered in San Francisco, California, within the Northern District of California.

68. Further, Elastic is informed and believes, and, on that basis, alleges that, floragunn made commercial use of its infringing Search Guard product by purposefully marketing and licensing that product to customers in the Northern District of California. By way of example, Elastic is informed and believes, and, on that basis, alleges that floragunn licensed its Search Guard software to: (1) PayPal Holdings, Inc., a company that, on information and belief, has its principal place of business in San Jose, California; (2) AppsCode, a company that, on information

and belief, has its principal place of business in San Leandro, California, for use in AppsCode's CubeDB software; (3) NVIDIA, a company that, on information and belief, has its principal place of business in Santa Clara, California; (4) Zuora, a company that, on information and belief, has its principal place of business in San Mateo, California; and (5) OpenTable, Inc., a company that, on information and belief, has its principal place of business in San Francisco, California

69. Additionally, over a span of several days in March 2019, floragunn actively promoted Search Guard to California entities and individuals while hosting a booth at a data security conference at the Moscone Center in San Francisco, California.

70. floragunn's marketing and commercial licensing of a directly competing product that infringes Elastic's copyright demonstrates an intent knowingly to harm Elasticsearch, Inc. a company with its principal place of business in Mountain View, California. It further shows that floragunn directed its infringing activities at the Northern District of California.

71. floragunn's infringement of Elastic's copyright has caused and continues to cause Elastic injury in the Northern District of California.

FLORAGUNN INDUCES THIRD PARTIES TO INFRINGE ELASTIC'S COPYRIGHTS

72. floragunn's marketing and distribution of infringing Search Guard software causes third party Search Guard users to incorporate code that infringes Elastic's copyrights in X-Pack and the X-Pack Kibana plugin. Those third parties therefore necessarily reproduce and use Elastic's proprietary X-Pack and/or X-Pack Kibana plugin code when they incorporate Search Guard into their adoptions of Elasticsearch, thereby infringing Elastic's copyrights.

73. Additional third parties have incorporated floragunn's infringing code into products and services they offer publicly. Elastic has investigated to identify third parties who have incorporated floragunn's infringing code into their products and services.

74. Among other infringing third party products and services that Elastic has identified, Amazon.com, Inc.'s and Amazon Web Services Inc.'s Open Distro for Elasticsearch ("Open Distro") and Amazon Elasticsearch Service ("AESS") offerings both contain and/or contained infringing code that originated with floragunn. Open Distro contains and/or contained infringing code related to floragunn's infringement of Elastic's copyrights in X-Pack and the X-

Pack Kibana plugin. AESS contains or contained infringing code related to floragunn's infringement of Elastic's copyrights in X-Pack. Rackspace US, Inc.'s ObjectRocket for Elasticsearch contains or contained infringing code related to infringement of Elastic's copyrights in X-Pack and the X-Pack Kibana plugin. And IBM Corporation's IBM Cloud Databases for Elasticsearch contains or contained infringing code related to infringement of Elastic's copyrights in X-Pack.

FLORAGUNN ATTEMPTS TO AVOID ENFORCEMENT OF UNITED STATES COPYRIGHT LAW

75. floragunn is undoubtedly aware that its conduct is unlawful. On the website for Search Guard, floragunn states that, just because "the source code of a piece of software is available for anyone to view and inspect," that "does not necessarily mean that the product is available at no cost, and it does not mean that it is solely a community product." floragunn goes on to warn "it is illegal to take our enterprise features into production without purchasing a license. ***This can lead to serious legal consequences, which can bring more harm and costs to a company*** . . . ." (emphasis added).

76. floragunn actively sought to avoid United States copyright law. On September 4, 2019, Elastic submitted Notices of Copyright Infringements under the DMCA to two websites that hosted floragunn's infringing code, GitHub, Inc. and Sonatype Inc. On or about September 11, 2019, GitHub and Sonatype removed floragunn's infringing code from their websites. Pursuant to the DMCA, floragunn then had the opportunity to submit counter notifications stating that Elastic's assertion that their content infringed Elastic's copyrights was mistaken. Submitting a counter notification could potentially have led to GitHub and Sonatype restoring floragunn's content to their websites. Elastic is not aware of any such counter notification by floragunn, and, on information and belief, floragunn did not take this opportunity to assert that its code did not infringe Elastic's copyright.

77. What floragunn did do, however, was switch the hosting of its infringing content to a different provider: Amazon Web Services, Inc. Amazon Web Services, Inc. maintains multiple offices in the Northern District of California.

78. On September 12, 2019, Elastic sent a Notice of Copyright Infringements under the DMCA to Amazon Web Services, Inc., and Amazon Web Services, Inc. removed floragunn's infringing code from its website on September 13, 2019. On information and belief, floragunn again did not take the opportunity to submit a counter notification and assert that its content did not infringe Elastic's copyrights. Once again, however, floragunn switched the hosting of its infringing content to a different provider.

79. It appears that floragunn's choice of the new host for its downloads was driven by a desire by floragunn to avoid takedowns required by the law of the United States. floragunn began hosting its infringing downloads through BlueAngelHost PVT. LTD. BlueAngelHost PVT. LTD. advertises "DMCA Ignored Hosting." It boasts that "***Purchasing USA-based hosting for a site that is not legal to be run in America is not a sensible thing to do. Offshore hosting can be helpful for less scrupulous businesses who wish to bypass local laws or regulations, particularly for issues like copyright law, which is also known as no DMCA hosting***" (emphasis added). BlueAngelHost PVT. LTD. lists a postal address in Serbia on its website and advertises data centers in Bulgaria, Russia, and the Netherlands.

80. Months after this lawsuit was filed, floragunn continues to include infringing code in Search Guard. In early October 2019, floragunn released new versions of its Search Guard products through web services run by GitLab Inc., Sonatype Inc., and floragunn's own website. These new versions purport to remove the infringing code that Elastic identified in its initial complaint. However, floragunn did not remove all instances of copying. The new versions of Search Guard continue to, at least, contain code that infringes Elastic's copyrights in its X-Pack Kibana plugin, as identified in this First Amended Complaint. Elastic continues to investigate floragunn's Search Guard code for instances of infringement and may identify further infringement.

FIRST CAUSE OF ACTION

Copyright Infringement

(17 U.S.C. § 101 *et seq.*)

81. Elastic incorporates by reference each of the allegations in the preceding paragraphs of this Complaint as if fully set forth here.

82. Before initiating this action, Elastic registered, effective August 14, 2019, versions 1.0.0 and 2.0.0 of Elasticsearch Shield (the predecessor name for X-Pack) and versions 5.0.0, 6.0.0, 6.2.0, 6.2.x, and 6.3.0 of X-Pack under Registration Numbers TX 8-762-996, TX 8-762-994, TX 8-762-975, TX 8-762-985, TX 8-762-987, TX 8-762-988, and TX 8-762-991, respectively. Elastic further registered, effective September 9, 2019, versions 1.1.1, 1.3.0, 2.0.0-beta1, and 2.0.0-beta2 of Elasticsearch Shield under Registration Numbers TX 8-773-254, TX 8-773-258, TX 8-773-261, and TX 8-773-263, respectively. Elastic additionally registered version 2.3.2 of the Kibana Shield plugin and versions 5.2.0, 5.3.1, 5.6.7, and 6.4.0 of the X-Pack Kibana plugin under Registration Numbers TX 8-796-945, TX 8-777-406, TX 8-777-412, TX 8-778-023, and TX 8-778-024, respectively, effective September 19, 2019; version 5.4.0 of the X-Pack Kibana plugin under Registration Number TX 8-796-010, effective November 4, 2019; and version 7.2.0 of the X-Pack Kibana plugin under Registration Number TX 8-796-013, effective October 31, 2019. Copies of those Certificates of Registration are attached as Exhibits E through V to this First Amended Complaint.

83. These works contain copyrightable subject matter for which copyright protection exists under the Copyright Act, 17 U.S.C. § 101, *et seq*. elasticsearch B.V. is the exclusive owner of all rights in these copyrighted works. Elasticsearch, Inc. holds the exclusive license from elasticsearch B.V. to enforce the copyright in and distribute copies of these works in, among other territories, the United States.

84. Through the actions described herein, floragunn has infringed and will continue to infringe Elastic's copyrights in the X-Pack and X-Pack Kibana plugin code by, at least, reproducing, preparing derivative works from, and distributing copies of those copyrighted works.

85. floragunn's infringing conduct alleged herein was and continues to be willful and with full knowledge of Elastic's rights in the copyrighted works, and that conduct has enabled floragunn to profit illegally from infringement.

86. Elastic is entitled to an injunction restraining floragunn, its officers, agents, employees, assigns, and all persons acting in concert with them from engaging in further infringement of Elastic's copyrights.

87. Elastic is entitled to recover from floragunn the damages it has sustained and will sustain as a result of floragunn's wrongful acts as alleged herein. Elastic is further entitled to recover from floragunn the gains, profits, and advantages it has obtained as a result of floragunn's wrongful acts. The full extent of Elastic's damages and the gains, profits, and advantages floragunn has obtained by reason of its aforesaid acts of copyright infringement cannot be determined at this time, but will be proven at trial. Further, Elastic is entitled to recover costs and reasonable attorneys' fees from floragunn as a result of the wrongful acts alleged herein.

SECOND CAUSE OF ACTION

Contributory Copyright Infringement

88. Elastic incorporates by reference each of the allegations in the preceding paragraphs of this Complaint as if fully set forth here.

89. floragunn's distribution of infringing Search Guard software induces, causes, encourages, and materially contributes to Search Guard users and third parties that incorporate Search Guard code into their products and services infringing Elastic's copyrights in the X-Pack and/or X-Pack Kibana plugin code by engaging in unauthorized reproduction and distribution of works containing Elastic's copyrighted material.

90. Elastic is informed and believes, and, on that basis, alleges that floragunn derived substantial financial benefit from Search Guard users' and third parties' infringement of Elastic's copyrights in X-Pack and/or the X-Pack Kibana plugin.

91. floragunn's marketing, commercial distribution of, licensing of, and profit from infringing Search Guard software shows that it knowingly, intentionally, willfully, and purposefully induced, caused, encouraged, and materially contributed to, and continues to

knowingly, intentionally, willfully, and purposefully induce, cause, encourage, and materially contributes to, Search Guard users' and third parties' infringement of Elastic's copyrights in X-Pack and/or the X-Pack Kibana plugin.

92. floragunn has the ability to prevent Search Guard users and third parties from infringing Elastic's copyrights in the X-Pack and X-Pack Kibana plugin code by omitting the infringing code from its Search Guard software product. However, floragunn has not prevented Search Guard users and third parties from infringing Elastic's copyrights in the X-Pack and X-Pack Kibana plugin code.

93. floragunn, through its knowing and intentional inducement, causation, encouragement, and material contribution to the infringement of Elastic's copyrights in the X-Pack and X-Pack Kibana plugin code by Search Guard users and third parties, is committing and/or is contributorily and vicariously liable for the acts of infringement by Search Guard users and third parties. Each act of infringement that floragunn knowingly and intentionally induced, caused, encouraged, and materially contributed to is a separate and distinct act of infringement.

94. Elastic is entitled to an injunction restraining floragunn, its officers, agents, employees, assigns, and all persons acting in concert with them from actions inducing, causing, encouraging, or materially contributing to Search Guard users' and third parties' infringement of Elastic's copyrights.

95. Elastic is entitled to recover from floragunn the damages it has sustained and will sustain as a result of floragunn's acts inducing, causing, encouraging, or materially contributing to Search Guard users' and third parties' infringement of Elastic's copyrights. Elastic is further entitled to recover from floragunn the gains, profits, and advantages it has obtained as a result of its acts inducing, causing, encouraging, or materially contributing to Search Guard users' and third parties' infringement of Elastic's copyrights. The full extent of Elastic's damages and the gains, profits, and advantages floragunn has obtained by reason of its aforesaid acts of copyright infringement by Search Guard users and third parties cannot be determined at this time but will be proven at trial. Further, Elastic is entitled to recover costs and reasonable attorneys' fees from

floragunn as a result of the acts inducing, causing, encouraging, or materially contributing to Search Guard users' and third parties' infringement of Elastic's copyrights alleged herein.

PRAYER FOR RELIEF

Elastic prays for judgment as follows:

1. For permanent injunctive relief, including an order restraining and enjoining floragunn and third parties using Search Guard and Search Guard code from further infringement of Elastic's copyrights, specifically:

   a. that floragunn and third parties using Search Guard products and code, as well as any successor entities, directors and officers, agents, servants, employees, assigns, and all other persons acting in active concert or privity or in participation with them, and each of them, be enjoined from continuing to market, offer, sell, dispose of, license, lease, transfer, display, advertise, reproduce, develop or manufacture infringing Search Guard software and any works derived or copied from infringing Search Guard software, or to participate or assist in any such activity;

   b. that floragunn and third parties using Search Guard products and code, as well as any successor entities, directors and officers, agents, servants, employees, assigns, and all other persons acting in active concert or privity or in participation with them, be enjoined from directly or indirectly infringing Elastic's copyrights in X-Pack and the X-Pack Kibana plugin;

   c. that floragunn and third parties using Search Guard products and code, as well as any successor entities, directors and officers, agents, servants, employees, assigns, and all other persons acting in active concert or privity or in participation with them, be enjoined to return to Elastic any originals, copies, facsimiles, or duplicates of Search Guard, any works derived or copied from Search Guard in their possession, custody, or control that are shown to infringe any Elastic copyright;

d. that floragunn and third parties using Search Guard products and code be enjoined to deliver upon oath, to be impounded during the pendency of this action, and for destruction pursuant to judgment herein, all originals, copies, facsimiles, or duplicates of Search Guard, any works derived or copied from Search Guard in their possession, custody, or control that are shown to infringe any Elastic copyright;

2. For compensatory damages against floragunn in an amount to be determined at trial;

3. For floragunn's profits obtained as a result of its infringing conduct, including but not limited to all profits from sales and other exploitation of Elastic's copyrighted material and any products, works, or other materials that include, copy, are derived from, or otherwise embody the copyrighted material, or in the Court's discretion, such amount as the Court finds to be just and proper;

4. For attorneys' fees and costs of suit incurred herein;

5. For interest, including pre-judgment and post-judgment interest, on the forgoing sums; and

6. For any other relief that the Court deems appropriate.

JURY DEMAND

Elastic demands a jury trial for all issues so triable.

Dated: November 26, 2019

DAVID R. EBERHART
JAMES K. ROTHSTEIN
O'MELVENY & MYERS LLP

By: /s/ David R. Eberhart
David R. Eberhart

Attorneys for Plaintiffs
ELASTICSEARCH, INC. and
ELASTICSEARCH B.V.